sensors

MDPI

Article

SemImput: Bridging Semantic Imputation with Deep Learning for Complex Human Activity Recognition

Muhammad Asif Razzaq [1], Ian Cleland [2], Chris Nugent [2] and Sungyoung Lee [1,*]

[1] Ubiquitous Computing Lab, Department of Computer Engineering, Kyung Hee University, Seocheon-dong, Giheung-gu, Yongin-si, Gyeonggi-do 446-701, Korea; asif@khu.ac.kr
[2] School of Computing, Ulster University, Jordanstown BT37 0QB, Northern Ireland, UK; i.cleland@ulster.ac.uk (I.C.); cd.nugent@ulster.ac.uk (C.N.)
* Correspondence: sylee@oslab.khu.ac.kr; Tel.: +82-31-201-2514

Received: 28 March 2020; Accepted: 8 May 2020; Published: 13 May 2020

Abstract: The recognition of activities of daily living (ADL) in smart environments is a well-known and an important research area, which presents the real-time state of humans in pervasive computing. The process of recognizing human activities generally involves deploying a set of obtrusive and unobtrusive sensors, pre-processing the raw data, and building classification models using machine learning (ML) algorithms. Integrating data from multiple sensors is a challenging task due to dynamic nature of data sources. This is further complicated due to semantic and syntactic differences in these data sources. These differences become even more complex if the data generated is imperfect, which ultimately has a direct impact on its usefulness in yielding an accurate classifier. In this study, we propose a semantic imputation framework to improve the quality of sensor data using ontology-based semantic similarity learning. This is achieved by identifying semantic correlations among sensor events through SPARQL queries, and by performing a time-series longitudinal imputation. Furthermore, we applied deep learning (DL) based artificial neural network (ANN) on public datasets to demonstrate the applicability and validity of the proposed approach. The results showed a higher accuracy with semantically imputed datasets using ANN. We also presented a detailed comparative analysis, comparing the results with the state-of-the-art from the literature. We found that our semantic imputed datasets improved the classification accuracy with 95.78% as a higher one thus proving the effectiveness and robustness of learned models.

Keywords: activity recognition; unobtrusive sensing; BLE; proximity; ontologies; semantic imputation; segmentation; neural network

1. Introduction

Over the past few decades, a rapid advancement has been observed in pervasive computing for the assessment of cognitive and physical well-being of older adults. For this purpose, monitoring of Activities of Daily Living (ADLs) is often performed over extended periods of time [1]. This is generally carried out in intelligent environments containing various pervasive computing and sensing solutions. Recognition of ADLs has been undertaken across a wide variety of applications including cooking, physical activity, personal hygiene, and social contexts. Generally, solutions for recognizing ADLs are underpinned with rule-based or knowledge-driven supported by conventional Machine Learning (ML) algorithms [2,3]. In such environments, the embedded or wireless sensors generate high volumes of streaming data [4], which in a real world setting can contain huge amounts of missing values or duplicate values [5]. Such noisy and imprecise data may lead to one of the major causes of an erroneous classification or imprecise recognition. Conversely, several challenges also exist while coping with missing values hence an efficient mechanism for imputation of the sensory data are thus

required. Issues in missing data become even more difficult when considering multimodal sensor data to recognize real-time complex ADLs. In this case, some of the sensors may generate continuous streams of data whilst others generate discrete streams [6].

Several statistical-based approaches are reported in the literature to deal with missing values. The majority of these propose data imputation solutions, the nature of which can vary depending on the size of the actual data and the number of missing values [7]. Most of them, however, use model-based imputation algorithms i.e., likelihood-based or logistic regression to encounter the missing values. The impact of imputation is determined by the classification performance, which may lead to biased parameter estimates, as most of the ML classifiers deal with the missing information implicitly. For this reason, complications whilst handling missing sensor states is still considered to be a non-trivial problem [8]. An appropriate strategy is therefore needed to improve the quality of data imputation with minimal computational efforts. Current approaches must also address data imputation in multimodal sensor streams, which not only improves the recognition performance but also increases overall robustness of the applications [9,10].

Despite the gain in statistical power, more recently, ontology-based modeling and representation techniques have been introduced [11]. These ontological models can discover, capture, encode rich domain knowledge, monitor patterns of ADLs, and provide heuristics in a machine-processable way [12,13]. Ontologies represent rich structured hierarchical vocabularies and can be used to explain the relations amongst concepts or classes. The coded knowledge is made accessible and reusable by separating sub-structural axioms, rules and conjunctions among the concepts [14]. In addition to separation logic, use of a query language, SPARQL also provides support for disengaging these semantics and assertions for interpreting any rule-based complex activities [15]. In work by Amador et al. [16], the authors used SPARQL for retrieving class entities and their types, which were later transformed into vector form before using deep learning approaches. Similarly, Socher et al. [17] have bridged neural networks with an ontological knowledge-base for the identification of additional facts. Only a limited amount of work, however, has been undertaken to account for semantic imputation using ontological models and SPARQL [18].

Moreover, the usability of semantic imputation and feature extraction using ontological methods in combination with deep neural networks for recognizing complex activities remains to be investigated. Previous studies have not provided a comprehensive analysis on the impact of imputation on the classification accuracy. To this end, we present research proving the applicability of semantic imputation for missing sensors and their states on activity classification in a controlled environment using deep-learning based Artificial Neural Networks (ANNs). This combination of semantic imputation with neural networks in a supervised learning method using public datasets not only increases accuracy, but also reduces the complexity of training data. The presented work is, to the best of our knowledge, the first to exploit ontologies, semantic imputation, and neural networks.

The key objectives being addressed in this study are to: (1) design and development of a practical scheme for modeling time-series data into an ontology, (2) perform semantic data expansion using the semantic properties, (3) identify suitable semantic data imputation measure, (4) design and train an effective deep learning model for Human Activity Recognition (HAR), and (5) undertake a comparative analysis using public datasets with each having different rates of missing data and imputation challenges.

The rest of the paper is structured as follows: Section 2 presents the problem formulation and key definitions. Section 3 elaborates on the structure of our proposed framework. In Section 4, we report the experimental evaluations and provide a comparative analysis using public datasets. Finally, Section 5 draws the conclusion and presents future work.

2. Problem Statement

In this section, we first introduce key definitions, which are carried throughout the paper. These definitions are necessary for understanding concepts referred to in this paper. Later, a robust illustrative example is presented to represent the research problem for HAR referred in this study.

2.1. Some Definitions

In this section, we first give preliminary definitions of problems that the methodology aims to address. Laterally, we introduce the notion of Semantic imputation.

Definition 1. *(Formal Notation) Let* $\{\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_n\}$ *be the set of multimodal sensory data of the form* $(p \times q)$ *matrices modeled over the domain ontologies* $\{\mathcal{O}_1, \mathcal{O}_2, \ldots, \mathcal{O}_n\}$ *respectively, where p represents the number of observations for q concepts (variables).*

Definition 2. *(Training Tuples) Let* $T_d = \{t_1, ..., t_p\}$ *be the set of training tuples for dataset* $\mathcal{D}_n$ *containing missing attributes or their values. Let* t_m *is a tuple with q attributes* $\{A_1, \ldots A_q\}$*, which may have one or more missing attributes or its value where* $t_m \in T_d$*. Let* t_{ma} *be the missing attribute A and* t_{mv} *be the missing value on attribute A where* $A \in A_q$*. Given a candidate imputed set,* $t_m = \bigcup_1^m (t_{ma} \cup t_{mv})$ *for a possible missing attributes or its value for* t_m*.*

Definition 3. *(Ontology) A core ontology is a structure* $\mathcal{O} := (C, \leq_c, R, \sigma, \leq_r)$ *consisting of two disjoint sets concept identifiers 'C' and relation identifiers 'R', a partial order* $\leq_c$ *on C, called concept hierarchy or taxonomy, a function* σ *representing signature, and a partial order* $\leq_r$ *on R defining relation hierarchy.*

Definition 4. *(Ontology-based Tuples) Given* o_k *and* o_l *in* $\mathcal{O}$*,* (o_k, o_l) *is called an ontology-based tuple, if and only if: (1)* $\exists\ A, B \in C \mid o_k \in A$ *and* $o_l \in B$*; (2)* $A \mapsto B$*; and (3)* $\lambda_{o_k}(o_l) \leq \gamma$*.*

Definition 5. *(Knowledge-base)A Knowledge Base* $\mathcal{K}$ *is conceptually referred to a combination of intentional terminologies TBox (*$\mathcal{T}$*) part and extensional assertion ABox (*$\mathcal{A}$*) part modeled over an ontology* $\mathcal{O}$*.* $\mathcal{T}$ *includes concept modeling and the relations in ontology* $\mathcal{O}$ *and* $\mathcal{A}$ *includes concept instances and roles.*

Definition 6. *(Conjunctive Query) Conjunctive queries* $\mathcal{Q}$ *enable answers by identifying attributes or their values, which are rewritten as*

$$\forall \bar{A} \bar{R}(\bar{A}, \bar{C}_k) \wedge not(\bar{N}(\bar{A}, \bar{C}_k)) \tag{1}$$

where $\bar{A}$ *represents vector of attributes* $(A_1, \ldots, A_q)$*, vectors of concept instances* $\bar{C}_k$*, conjoined predicates (relations)* $\bar{R}$*, and a vector of disjoined predicates (relations)* $\bar{N}$*.*

2.2. Problem Formulation: Semantic Imputation

A Knowledge Base is a consistent structure $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, and we revise the Abox $\mathcal{A}$ to $\mathcal{A}^I$ such that $\mathcal{K} = (\mathcal{T}, \mathcal{A}^I)$ should also be consistent:

$$\mathcal{A}^I = \mathcal{A} \cup \mathcal{I}(A_m) \qquad \text{since } (\mathcal{A}_m = D_n \setminus \mathcal{A}) \tag{2}$$

$$\mathcal{I}(A_m) = \mathcal{I}_{SS}(A_m) + \mathcal{I}_{SI}(A_m) + \mathcal{I}_L(A_m) \tag{3}$$

where A_m represents missing attributes or their values and $\mathcal{I}_{SS}(A_m)$, $\mathcal{I}_{SI}(A_m)$, $\mathcal{I}_L(A_m)$ measure structural-based, instance-based and longitudinal imputations for missing attributes and their values, respectively.

Hence, we define our problem in a 4-tuple ($\mathcal{D}$, $\mathcal{K}$, $\mathcal{Q}$, $\mathcal{I}$) such that $\mathcal{D}$ denotes the input data, modeled over the ontology $\mathcal{O}$ having assertion set $\mathcal{A}$ which are retrieved using conjunctive queries $\mathcal{Q}$ with the results used to perform semantic imputation $\mathcal{I}(A_m)$ introducing improved assertions

$\mathcal{A}^I$. We ensure that, during the whole process, $\mathcal{K}$ remains consistent with the addition of imputed assertions $\mathcal{A}^I$.

2.3. Preliminaries of Sensing Technologies

In this section, we describe the nature of available HAR public datasets $\mathcal{D}_n$ with underlying sensing technologies. These can be differentiated into two broad categories of *unobtrusive* and *obtrusive* activity sensing based on the wearables and data sources. We, therefore, provide a brief description of both categories using *UCamI* [19], *Opportunity* [20], and *UCI-ADL* [21] public datasets for their distinct sensing functionalities, signal type, sampling frequencies, and protocols.

2.3.1. Unobtrusive Sensing

Unobtrusive sensing enables continuous monitoring of activities and physiological patterns during the daily life of the subject. These wearables most often involve binary sensors (BinSens), PIR sensors, and pressure sensors embedded within smart objects or the ambient environment. *BinSens* generate an event stream comprising of binary values, working on the principles of the Z-Wave protocol. Such protocols are implemented through some unobtrusive wireless magnetic sensors. This can be explained through the *Prepare breakfast* example in Figure 1. For 'Pantry', 'Refrigerator', and 'Microwave' objects, *Open* state means magnets are detached and they are in use, whereas *Close* state shows they are not in use. The inhabitant's movements are recorded at a sample rate of 5 Hz, using the ZigBee protocol implemented in 'PIR sensors' such as the 'Sensor Kitchen Movement' [22]. It also produces binary values with *Movement* or *No Movement*. The presence of an inhabitant on the 'Sofa', 'Chair', and 'Bed' objects are collected via the Z-Wave sensing protocol, implemented through the 'Textile Layer Sensors', which produce binary values *Present* or *Not present*. Similarly, a continuous stream of data are also observed for unobtrusive spatial data gathered through the suite of capacitive sensors installed underneath the floor.

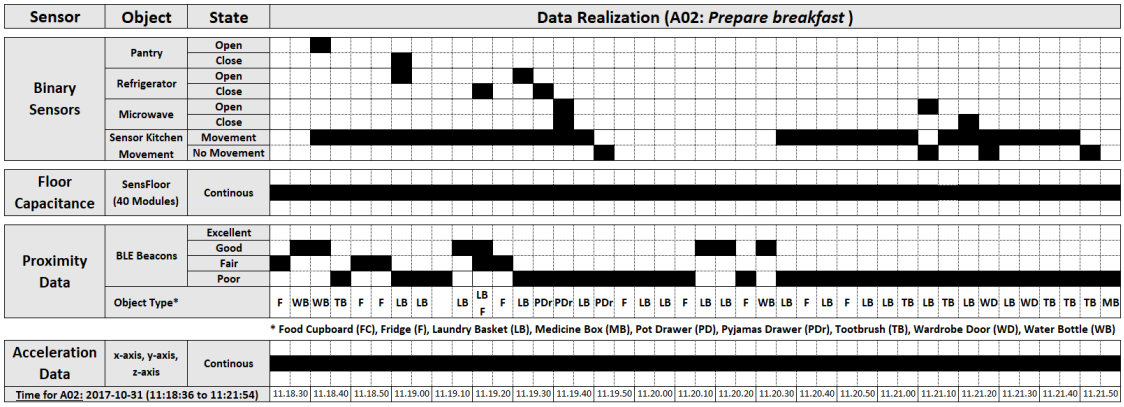


Figure 1. Time series analysis for example *Prepare breakfast* in *UCamI* dataset [19].

The dataset generated through the *BinSens* is of a challenging nature as the duration of the generated stream may be instantaneous, lasting for a few seconds or may continue for hours. As shown in Figure 1, filling the gaps between two states for *BinSens* is of a challenging nature since every *BinSens* has a different operation nature and state transition time depending on the activities performed.

2.3.2. Obtrusive Sensing

The proximity data from the Bluetooth Low Energy (BLE) beacons is collected through an android application installed on the smart-watch at a sample rate of 0.25 Hz [22]. BLE beacons are measured through RSSI. The value of the RSSI is higher if there is the smaller distance between an object and the smart-watch and vice versa. BLE beacons are used for 'Food Cupboard', 'Fridge', 'Pot Drawer', etc.,

for the *Prepare breakfast* activity example in Figure 1. Ambulatory motion is represented by *Acceleration* data, which is again gathered through the android application installed on the smart-watch. The 3D acceleration data are collected in a continuous nature using a sampling frequency of 50 Hz. Such acceleration data [20] is also measured through body-worn sensors, object sensors and ambient sensors, which measure 3D acceleration using inertial measurement units, 3D acceleration with 2D rate of turn and 3D acceleration with multiple switches, respectively.

3. Methodology

In this section, we demonstrate the proposed methodology, overall functional architecture and workflow in Section 3.1. An ontology model to represent the activities is presented in Section 3.2 and a detail of specially designed SPARQL queries for semantic segmentation in Section 3.3. Ontology-based complex activities identification and conjunction separation for semantic data expansion is explained in Section 3.4. An algorithm to perform semantic imputation is then described in Section 3.5. Lastly, the classification method describing HAR using DL based ANNs is presented.

3.1. High-Level Overview of the SemImput Functional Framework

The presented work describes a layered Semantic-based Imputation (*SemImput*) framework, which supports an innovative means to synchronize, segment, and complete the missing sensor data. This is achieved by automatically recognizing the indoor activities within the smart environment. The architecture depicted in Figure 2 comprises of (a) *Data Sensing and Representation Layer* designed to capture data; (b) the *Semantic Segmentation Layer* segments the data based on the timestamps for over 1-second; (c) the *Semantic Expansion Layer* segregates the concurrent activities represented by separate features into a sensor event matrix; (d) the *Semantic Imputation Layer*, responsible to fill the missing data, sensor states, which are of periodic nature and provides continuity to the data by using the proposed strategies; (e) the *Semantic Vectorization* receives the filled sensor event matrix and generates vector sets; (f) and finally the *Classification Layer*, which uses a neural network to classify the augmented Semantic Vectors for evaluation purposes.

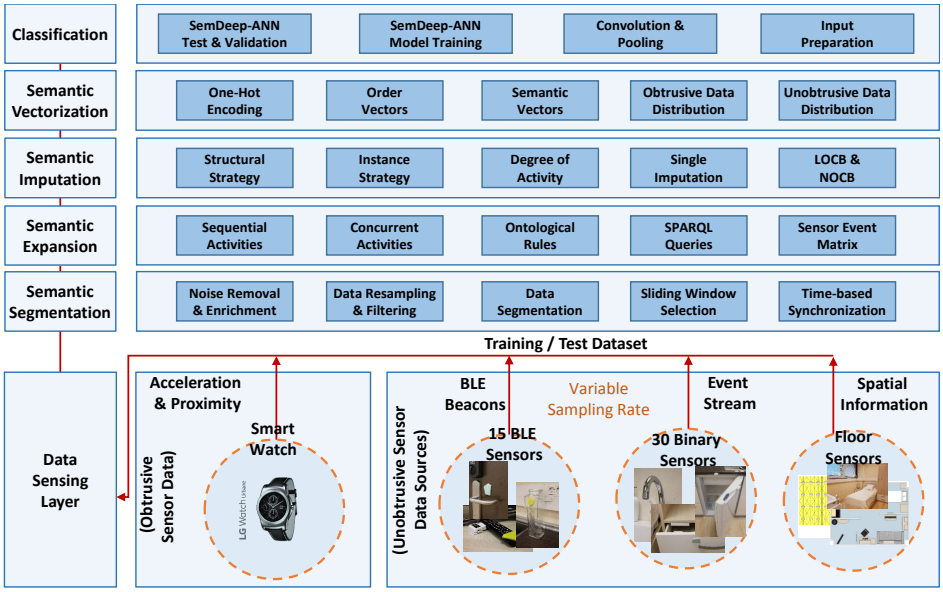


Figure 2. A detailed view of *SemImput* framework.

3.2. Data Sensing and Representation

The *Data Sensing and Representation* layer utilizes the sensor streams which are simulated over a dynamic sliding window. We used ontological constructs, which are derived through the data-driven techniques for representing sequential and parallel activities. This layer is encapsulated by the newly

modeled set of OWL2 Semantic Imputation Ontologies (*SemImputOnt*) to map sensory data. It models sensor streams, identifies patterns, and discovers the overlapping temporal relations in them. It supports generality in terms of data semantization [23], offers more expressiveness, and helps in decoupling the concurrent fragments of sensor data rather than using non-semantic models. It not only provides a basic model for representing the atomic and complex ADLs but also supports the expansion of dataset instances through the SPARQL queries.

3.2.1. Taxonomy Construction

We followed and utilized the data-driven techniques to model sensor streams for identifying complex concurrent sensor temporal state patterns. These state patterns become the basis for the parallel and interleaved ADLs, which are of static and dynamic nature as mentioned in Table 1. An ontology engineer utilizes the complete knowledge of involved sensors and the nature of the data produced by them. In addition, the core vocabulary required to model and design the *SemImputOnt* is obtained through the temporal patterns of sensor stream data, describing the complex ADL's main class definitions. The descendants of these main classes, however, have been described to model each sensor object, which generates discrete or continuous sensory data. These primitive classes are related to ADLs using "SensorStateObject" properties. These object properties such as *hasBinarySensorObject* shows the relationship between the ADL and the core sensor object defining its state. Again, the state is linked by a property *hasBinarySensorState* with *SensorStateObjects*. Similarly, the other obtrusive sensor objects have the properties *hasAccelerometer*, *hasBLESensor* with the *hasRSSI* data property. All these sensor objects define the ADL with open intervals without any prior knowledge of *Start-time* or *End-time* [1]. The temporal relations for each sensor object are obtained using object properties *hasStartTime* and *hasEndTime*.

How comprehensive *SemImputOnt* is at representing disjoint ADLs can be visualized and explained through an example of the activity *Breakfast* modeled in Figure 3. In this example, an ADL *Breakfast* is represented as a class. The ADL *Breakfast* is a descendant of the *Activities* class, defined as being an equivalent class relating to the instances of BinarySensorObject, BinarySensorState, Accelerometer, Devices, FloorCapacitance, BLESensors, and DaySession. This means that, to be a member of the defined class *Breakfast*, an instance of the *Activities* class must have a property of type *hasBinarySensorObject*, which relates to an instance of the *SensorKitchenMovement* class, and this property can only take as value an instance of the *SensorKitchenMovement* class. The instance of the *Activities* class must also have a property of type *hasBinarySensorState*, which relates to an instance of the *Movement* class, or the *NoMovement* class, and this property can only take as value an instance of one of them. The instance of the *Activities* class must also have a property of type *hasAccelerometer*, which relates to an instance of the *x* class, *y* class, and *z* class. This property must only relate to the instances of these three classes. The instance of the *Activities* class must also have a property of type *hasDevice*, which relates to an instance of the *Device1* class, and *Device2* class. This property must only relate to the instances of these two classes. The instance of the *Activities* class must also have a property of type *hasFloorCapacitance*, which relates to an instance of the *C1* class, *C2* class, *C3* class, *C4* class, *C5* class, *C6* class, *C7* class, and *C8* class. This property must only relate to the instances of these seven classes. The instance of the *Activities* class must also have a property of type *hasBLESensor*, which relate to an instance of the *Tap* class, *FoodCupboard* class, *Fridge* class, and *WaterBottle* class for this example. This property must only relate to the instances of these four classes and every class must also have a property *hasRSSI*, which relates to the instance of *RSSI* class. Moreover, the instance of the *Activities* class must also have a property of type *hasDaySession*, which relates to an instance of the *Morning* class and only to an instance of the *Morning* class. Thus, if an instance of the *Activities* class fulfills the seven existential restrictions on the properties *hasBinarySensorObject*, *hasBinarySensorState*, *hasAccelerometer*, *hasDevice*, *hasFloorCapacitance*, *hasBLESensor*, and *hasDaySession*, the instance will be inferred as being a member of the *Breakfast* class.

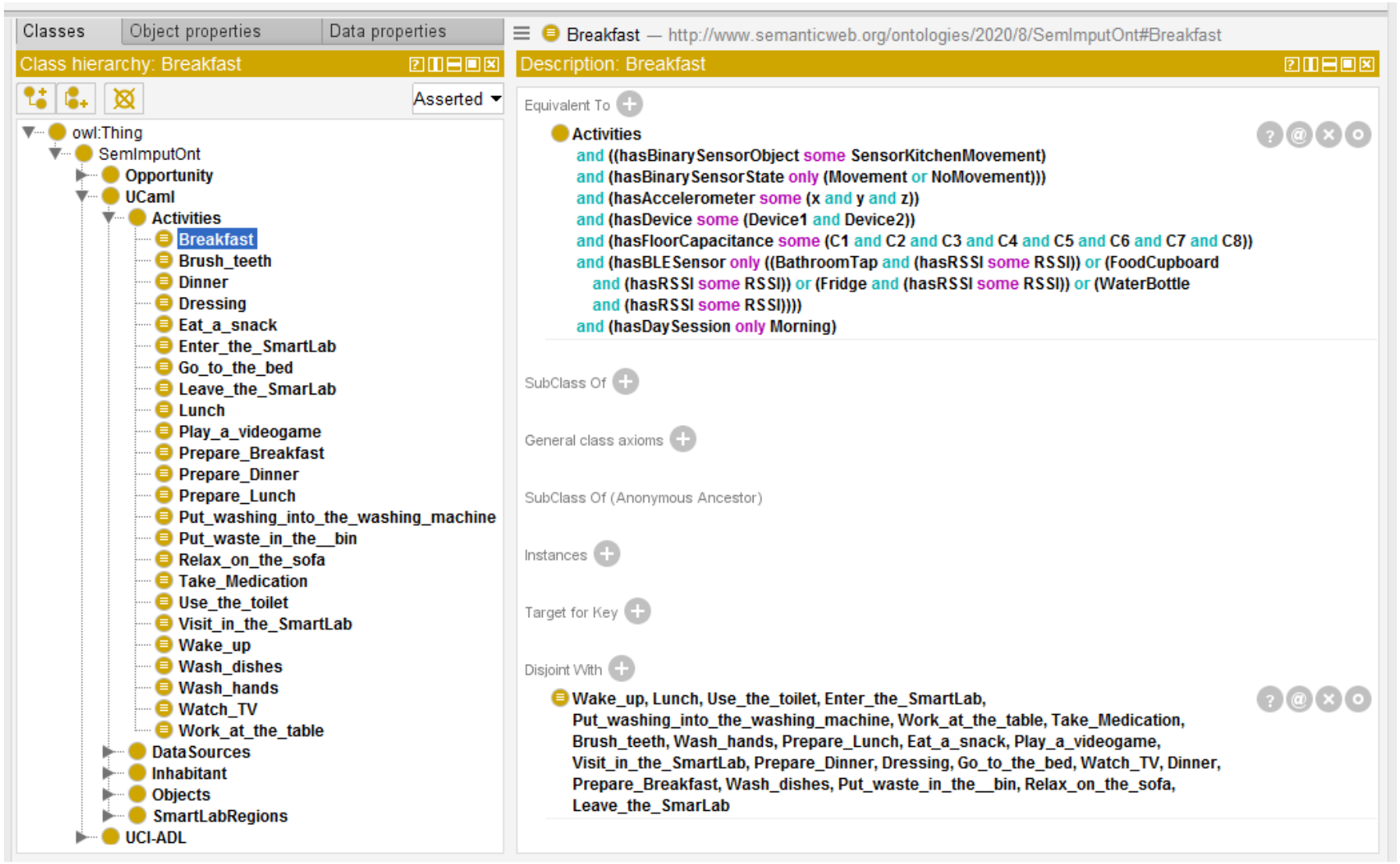


Figure 3. *SemImputOnt*: Class hierarchy with a definition axiom for the activity *Breakfast*.

Table 1. A list of activities, locations, and dependent sensor objects identified from UCamI dataset utilized for *SemImputOnt* constructs.

Type	ID	Activity Name	Location	Activity Dependencies Sensors' Objects
Static	Act01	Take medication	Kitchen	Water bottle, MedicationBox
Dynamic	Act02	Prepare breakfast	Kitchen, Dining room	Motion Sensor Bedroom, Sensor Kitchen Movement, Refrigerator, Kettle, Microwave, Tap, Kitchen Faucet
Dynamic	Act03	Prepare lunch	Kitchen, Dining room	Motion Sensor Bedroom, Sensor Kitchen Movement, Refrigerator, Pantry, Cupboard Cups, Cutlery, Pots, Microwave
Dynamic	Act04	Prepare dinner	Kitchen, Dining room	Motion Sensor Bedroom, Sensor Kitchen Movement, Refrigerator, Pantry, Dish, microwave
Dynamic	Act05	Breakfast	Kitchen, Dining room	Motion Sensor Bedroom, Sensor Kitchen Movement, Pots, Dishwasher, Tap, Kitchen Faucet
Dynamic	Act06	Lunch	Kitchen, Dining room	Motion Sensor Bedroom, Sensor Kitchen Movement, Pots, Dishwasher, Tap, Kitchen Faucet
Dynamic	Act07	Dinner	Kitchen, Dining room	Motion Sensor Bedroom, Sensor Kitchen Movement, Pots, Dishwasher, Tap, Kitchen Faucet
Dynamic	Act08	Eat a snack	Kitchen, Living room	Motion Sensor Bedroom, Sensor Kitchen Movement, Fruit Platter, Pots, Dishwasher, Tap, Kitchen Faucet
Static	Act09	Watch TV	Living room	RemoteControl, Motion Sensor Sofa, Pressure Sofa, TV
Dynamic	Act10	Enter the SmartLab	Entrance	Door
Static	Act11	Play a video game	Living room	Motion Sensor Sofa, Motion Sensor Bedroom, Pressure Sofa, Remote XBOX
Static	Act12	Relax on the sofa	Living room	Motion Sensor Sofa, Motion Sensor Bedroom, Pressure Sofa
Dynamic	Act13	Leave the SmartLab	Entrance	Door
Dynamic	Act14	Visit in the SmartLab	Entrance	Door
Dynamic	Act15	Put waste in the bin	Kitchen, Entrance	Trash
Dynamic	Act16	Wash hands	bathroom	Motion Sensor Bathroom, Tap, Tank
Dynamic	Act17	Brush teeth	bathroom	Motion Sensor Bathroom, Tap, Tank
Static	Act18	Use the toilet	bathroom	Motion Sensor Bathroom, Top WC
Static	Act19	Wash dishes	Kitchen	dish, dishwasher
Dynamic	Act20	Put washing into the washing machine	Bedroom, Kitchen	Laundry Basket, Washing machine, Closet
Static	Act21	Work at the table	Workplace	
Dynamic	Act22	Dressing	Bedroom	Wardrobe Clothes, Pyjama drawer, Laundry Basket, Closet
Static	Act23	Go to the bed	Bedroom	Motion Sensor bedroom, Bed
Static	Act24	Wake up	Bedroom	Motion Sensor bedroom, Bed

3.2.2. Concurrent Sensor State Modeling

The object properties introduced in *SemImputOnt* as an existential restriction support management of concurrent and sequential sensor states as explained in the *Breakfast* activity model example. These properties not only describe the hierarchy of sensor object states, and their actions by establishing object–data relationships but also support in augmenting the incomplete sensor sequences using SPARQL queries. Moreover, the relationship also supports, while generalizing data-driven rules as shown in the anonymous equivalent class for the activity *Breakfast*. These rules map sensor states in *SemImputOnt* to model an activity rather than tracking rigid sensor state patterns. These sensor state patterns are identified and linked to their respective timestamps using temporal datatype properties such as *hasStartTime* and *hasEndTime*. *SemImputOnt* comprehensively models sensor situations using sensor state concepts independently and concurrently by exploiting their relationships using *Allen's temporal operators* [15].

3.3. Semantic Segmentation

The Semantic Segmentation Layer in the *SemImput* framework describes the ontological operations to illustrate the modeling patterns of ADLs, by observing them in a sliding window. The first step is to retrieve and synchronize the non-segmented sensor state instances obtained from obtrusive and unobtrusive data sources along with their temporal information. We used a non-overlapping and static sliding time windows [24] approach, in which each sensor state is identified by a timestamp. For this, we used a set of 9 SPARQL-based query templates for retrieving and interpreting rules to deal with underlying temporal sensor state relations, as well as their structural properties. Moreover, the SPARQL queries require additional parameters in order to correlate, interpret, and aggregate sensor states within the endpoints of the sliding window [25]. Some of the initializing parameters include *start-time*, *end-time*, and a list of sensors within the sliding window identified based on the *start-time* and datatype properties. These parameters provide support for manipulating concurrent sensors states, which are expanded and imputed as illustrated in further sections. *SemImputOnt* is also used for validating temporal constraints and for the verification of property values within a sliding window [26]. The sensor state endpoints are retrieved through the following custom set of conjunctive ABox SPARQL queries $\mathcal{CQ}$ where ($cq_i \, \epsilon \, \mathcal{CQ}$) over the sliding time window:

- cq_1: Valid *Open* sensor state
- cq_2: Valid *Closed* sensor state
- cq_3: *Start-time* of *Next*, sensor state
- cq_4: Sensor having *Open* state within the sliding window
- cq_5: Sensor having *Closed* state within the sliding window

whereas the concurrent sensor states are retrieved through following SPARQL-based query templates, which are also coincidental at their:

- cq_6: *start-time* and still *Open* sensor states
- cq_7: *start-time* but *Closed* sensor states
- cq_8: *end-time* but still *Open* sensor states
- cq_9: *end-time* but *Closed* sensor states

The SPARQL query, cq_1, refers to the identifiers from the *SemImputOnt* retrieved instances, which are still active but are yet to be finished. These states are identified based on their initialization timestamps represented by the *start-time*. The query cq_2 retrieves *SemImputOnt* instances having both endpoints identified by *start-time* and *end-time*. The query cq_3 retrieves the *start-time* of the sensor initialization, which may deactivate and at the same time becomes active in a current sliding time window. The query cq_4 retrieves sensor state, which has just started in the sliding window; this query provides the *start-time*. The query cq_5, a specially designed query to monitor the sensor state, which is currently active in the sliding window and changes its states to *deactivation* or *off state*. This query

retrieves the *end-time* for such state transition. The query cq_6 retrieves active concurrent sensor states for more than one sensor, based on the *start-time* within the current sliding time window which is yet to finish. The query cq_7 on the other hand fetches the *start-time* for such concurrent sensors, which have *closed* states with valid *end-times*. Similarly, the queries cq_8 and cq_9 retrieve the active and inactive concurrent sensor states based on some *end-time* data value, respectively. The above-mentioned queries cq_3, cq_4, and cq_6 are responsible for initializing a separate thread to monitor and keep the track for sensor states which are to become inactive by identifying the *end-time*.

The segments returned through the SPARQL queries may be considered complete if they contain both the endpoints represented by dissimilar sensor states. If one of the end points goes missing, however, the segment becomes anomalous or erroneous in the sensor stream data. Such erroneous behavior is identified by using semantic data expansion and resolved through the semantic imputation.

3.4. Semantic Data Expansion

The proposed set of *SemImputOnt* models sensor objects (concepts and properties) and their states (instances) from the segmented D_n datasets. It not only maps sensor streams but also captures structure, preserving the associations within the sensor state instances using a data-driven approach. A structure-preserving transformation encompasses each sensor object, their associations, and subsumptions relating to different concurrent activities [27]. These preserved semantics and associations are separated by understanding the complex activity structures. The separation process includes conversions of these semantics into distinct columns while conjunctions in between them provide essential existential conditions for representing activities in a matrix.

3.4.1. Ontology-Based Complex Activity Structures

To encode more detailed structure, the *SemImputOnt* uses primitive and defined concepts with value-restriction and conjunctions as concept-forming operators. These value restrictions are enforced through classifiable attributes (roles) and non-classifiable attributes (non-definitional roles) to model HAR datasets. In *SemImputOnt*, primitive-concepts (Activities) provide necessary conditions for membership, whereas defined concepts (Sensors, Objects, Data sources) provide both necessary and sufficient conditions for membership as mentioned below:

$$\mathcal{A} \sqsubseteq C; \tag{4}$$

$$\mathcal{A} \equiv C; \tag{5}$$

where $\mathcal{A}$ is any *Activity* name, and C defines a primitive concept or a defined concept as mentioned in Equations (4) and (5), respectively. These concepts are used to form an expression, which can be either a sensor state, or conjunction of sensor states with or without a value-restriction as described below:

$$C \rightarrow A_1; C \rightarrow (\forall R.A_2 \sqcap \exists R); C \rightarrow C_1 \sqcap C_2 \tag{6}$$

Here, A_1, A_2 are attribute, R is a conjoined predicate, and C_1, C_2 are concept instances forming expressions.

Utilizing the Description Logic (DL) notations, an example of *Breakfast Activity* from *UCamI* dataset can be described in DL expression as:

Breakfast $\equiv$ *Activities* $\sqcap$ $\exists$ *hasBinarySensorObject.SensorKitchenMovement* $\sqcap$ $\forall$ *hasBinarySensorState.(Movement* $\sqcup$ *NoMovement)* $\sqcap$ $\exists$ *hasAccelerometer.(x* $\sqcap$ *y* $\sqcap$ *z)* $\sqcap$ $\exists$ *hasDevice.(Device1* $\sqcap$ *Device2)* $\sqcap$ $\exists$ *hasFloorCapacitance.(C1* $\sqcap$ *C2* $\sqcap$ *C3* $\sqcap$ *C4* $\sqcap$ *C5* $\sqcap$ *C6* $\sqcap$ *C7* $\sqcap$ *C8)* $\sqcap$ $\forall$ *hasBLESensor.(Tap* $\sqcap$ $\exists$ *hasRSSI.RSSI* $\sqcup$ *FoodCupboard* $\sqcap$ $\exists$ *hasRSSI.RSSI* $\sqcup$ *Fridge* $\sqcap$ $\exists$ *hasRSSI.RSSI* $\sqcup$ *WaterBottle* $\sqcap$ $\exists$ *hasRSSI.RSSI)* $\sqcap$ $\forall$ *hasDaySession.Morning*

whereas the same activity *Breakfast* using the DL attributes from *UCI-ADL* dataset is described as:

Breakfast ≡ *UCI-ADL* ⊓ ∃ *hasPlace Kitchen* ⊓ ∀ *hasPlace Kitchen* ⊓ ∃ *hasSensorLocation (Cooktop* ⊔ *Cupboard* ⊔ *Fridge* ⊔ *Microwave* ⊔ *Seat* ⊔ *Toaster)* ⊓ ∀ *hasSensorLocation (Cooktop* ⊔ *Cupboard* ⊔ *Fridge* ⊔ *Microwave* ⊔ *Seat* ⊔ *Toaster)* ⊓ ∀ *hasSensorType (Electric* ⊔ *Magnetic* ⊔ *PIR* ⊔ *Pressure)*

In both the expressions, the activity *Breakfast* is represented by different concept attributes modeled into their corresponding ontologies in the *SemImputOnt*. It is evident that this activity is represented by different sets of underlying ontological concepts depending upon the nature of sensors deployed for acquiring the datasets for that activity. Keeping the same definition of each activity represented by different underlying constructs may result in recognition performance degradation. For this reason, they are defined separately, as the focus of the study is to fill in the gaps for missing sensor states.

The primitive concepts are mapped into partial concepts using Web Ontology Language (OWL), which are encoded with *rdfs:subClassOf* construct (Equation (4)). In addition, the defined concepts are mapped to complete concepts in OWL, which are encoded as class equivalence axioms represented as *owl:equivalentClass* (Equation (5)). The concept names and concept conjunctions are mapped to class names and class intersections in OWL, respectively, whereas roles are mapped with object properties. These primitive and defined concepts definitions map the data instances into *SemImputOnt* models for representing complex activities.

3.4.2. Conjunction Separation

The concepts expressed in the DL for *Breakfast* definition uses conjunctions for relating the sensor state events [28]. The *Breakfast* equivalent class forming a complex activity with the involvement of several *Class* concepts, relationships (object & data properties), and data instances. All the involved *Class* concepts coupled with conjunctions defining the *Activity* equivalent classes are transformed into independent entities by separating them based on involved conjunctions [14]. Conjunction separation emphasizes the idea of concept $(\varphi, \psi, \omega, \chi \dots)$ separation over the intention I such as:

$$\models I\left(\varphi \wedge \psi \wedge \omega \wedge \chi \dots\right) \rightarrow I\left(\varphi\right) \wedge I\left(\psi\right) \wedge I\left(\omega\right) \wedge I\left(\chi\right) \dots \tag{7}$$

These independent entities are transformed into multi-dimensional vectors representing the features from all sensor states for a particular activity w.r.t. associated timestamps. The size of the multi-dimensional vector may vary for each activity based on the conjunctive class concepts learned through the data modeled over *SemImputOnt*.

3.4.3. Feature Transformation

The predicates separated in the previous step produces a row vector identified by a single activity label, whereas column represents the class concepts with states as an instance. These predicates in the feature space represent activities along with the timeline. These features ensure the reliability of activities through mappings with the *SemImputOnt* [12,28]. In our case, *SemImputOnt* supports essential properties while generating and validating the data into ABox $\mathcal{A}$ features as provided using an example from the *UCamI* dataset.

$$\mathcal{A}_n \leftarrow \{BinSens_1, BinSens_2, \dots BinSens_{30}, BLE_1, \dots BLE_{15}, C_1, C_2, \dots C_8, x, y, z\} \tag{8}$$

where $n = \{1, 2, \dots, 24\}$, *BinSens* can have one of the states at a unit time T_{1sec} from {*Open*, *Close*, *Present*, *No present*, *Pressure*, *No Pressure*, *Movement*, *No Movement*}. These state mappings result into a matrix representing each row with a single activity and every column with *Class* concepts. Each of the separated concept supports modification of one segment independent of the others column-wise.

3.5. Semantic Data Imputation

The resulting n-dimension feature vector matrix has missing sensor states (*Null*), which lead to the loss in efficiency for the activity classification model. Such losses can be dealt with suitable imputation

techniques, which enriches the expanded data semantically by filling in the missing sensor states. We propose a *Semantic Imputation* algorithm to capture the temporal missing sensor states semantically and perform an overall feature vector matrix enrichment [29]. We adapt two similarity-based methods and a time-series longitudinal imputation strategy to assess similarity of the concepts $\mathcal{T}$ and instances $\mathcal{A}$ for imputation $\mathcal{I}(A_m)$ as described in Algorithm 1.

Algorithm 1 Semantic Imputation Using $\mathcal{I}_{SS}(A_m)$, $\mathcal{I}_{SI}(A_m)$, and $\mathcal{I}_L(A_m)$ through SPARQL Queries

Input: Incomplete Segmented Data $A_m, \mathcal{A}, D_{seg}$

Output: Complete Data with Imputation A_m^{Imp} ▷ Segmented Imputed Dataset.

procedure SEMANTICIMPUTATION
  for all timestamp t = 1 to T **do**
    function $ImputeBinSens(A_m, CQ, A, T)$ ▷ $BinSens_{attrib}$ with their state imputation
      for $(cq_i \epsilon \mathcal{CQ})$ **do**
        $BinSens_{Attrib} \leftarrow execute(cq_i).filter(BinSens, A_m)$ ▷ using SPARQL Queries
        $BinSens_{Target} \leftarrow execute(cq_i).filter(BinSens_{Attrib}, \mathcal{T})$
        $A_{BS_{att}} \leftarrow BinSens_{Attrib}$
        $A_{BS_{tar}} \leftarrow BinSens_{Target}$
        $max(\mathcal{I}_{SS}) \leftarrow Compute\ \mathcal{I}_{SS}(A_{BS_{tar}}, A_{BS_{att}})$ ▷ Equation (10)
        $A_{BS_{att}} \leftarrow A_{BS_{att}} \cup (A_{BS_tar} \setminus A_{BS_{att}})$ ▷ Update missing BinSens Attribute
        $BinSens_{mappings} \leftarrow retrieve.mappingsLists(BinSens_{LOCF}, BinSens_{NOCB})$
        while $A_{BS_{att}}(state) = \phi$ **do** ▷ Load Updated *BinSens* attributes
          if $(A_{BS_{att}}$ in $BinSensList_{LOCF})$ **then** ▷ based on *BinSens* characteristics
            $A_{BS_{state}} \leftarrow execute(cq_i).retrieveLastState.(A_{BS_{att}})$
            $A_{BS} \leftarrow \mathcal{I}_L(A_{BS_{att}}, A_{BS_{state}})$
          else if $(A_{BS_{att}}$ in $BinSensList_{NOCB})$ **then**
            $A_{BS_{state}} \leftarrow execute(cq_i).retrieveNext, State.(A_{BS_{att}})$
            $A_{BS} \leftarrow \mathcal{I}_L(A_{BS_{att}}, A_{BS_{state}})$
      Return Imputed A_{BS}
    function $ImputeProximity(A_m, CQ)$ ▷ Imputation for Proximity Sensors and their values
      for $(cq_i \epsilon \mathcal{CQ})$ **do**
        $A_{Prox} \leftarrow execute(cq_i).filter(Proximity, A_m)$
        $Prox_{max} \leftarrow maxValue(A_{Prox})$
        $A_{Prox} \leftarrow Update\ A_{Prox}(Prox_{max})$
      Return Imputed A_{Prox}
    function $ImputeFloor(A_m, CQ, A)$ ▷ Imputation for Floor sensors and their values
      for $(cq_i \epsilon \mathcal{CQ})$ **do**
        $A_{mfloor} \leftarrow execute(cq_i).filter(Floor, A_m)$
        $A_{tfloor} \leftarrow execute(cq_i).filter(A_{mfloor}, A)$
        $mean(floortuples) \leftarrow Compute\ \mathcal{I}_{SI}(A_{tfloor}, A_{mfloor})$ ▷ Equation (13)
        $A_{floor} \leftarrow$Update $A_{mfloor} \cup mean(floortuple)$ ▷ update using mean for tuples
      Return Imputed A_{floor}
    function $ImputeAccelerometer(A_m, CQ, A)$ ▷ Imputation for accelerometer values
      for $(cq_i \epsilon \mathcal{CQ})$ **do**
        $A_{mAcc} \leftarrow execute(cq_i).filter(Acc, A_m)$
        $A_{tAcc} \leftarrow execute(cq_i).filter(A_{mAcc}, A)$
        $mean(acctuples) \leftarrow Compute\ \mathcal{I}_{SI}(A_{tAcc}, A_{mAcc})$
        $A_{Acc} \leftarrow$Update $A_{mAcc} \cup mean(acctuples)$ ▷ update using mean for last 10 tuples
      Return Imputed A_{Acc}
    $A_m^{Imp} \leftarrow A_{BS} \parallel A_{Prox} \parallel A_{floor} \parallel A_{Acc}$
    increment t by 3 sec

3.5.1. Structure-Based Imputation Measure

The structural patterns in TBox ($\mathcal{T}$) are identified and exploited using SPARQL queries over the *SemImputOnt*. These queries could retrieve $\mathcal{T}$ assertions based on the query criteria to measure semantic similarity with target activity patterns. However, choosing a suitable pattern from target activities and selecting the appropriate sensor state to fill in the missing ones is addressed through structure-based similarity measure. We define structural similarity function for a target set of description A_n and activity A_m with missing attributes to identify maximum probability as:

$$Sim_{ss} : A_n \times A_m \longmapsto [0 \ldots 1] \tag{9}$$

It returns semantically equivalent sensor states where the child nodes for two concepts are similar [30]. We use the *Tanimoto* coefficient between A_n and A_m for measuring the structural similarity. A_n gives the binary description for the involved sensors and A_m are the available sensor predicates for the activity with missing predicates mentioned below:

$$\begin{aligned} \mathcal{I}_{SS}(A_m) &= Sim_{ss}(A_n, A_m) \\ &= \frac{\sum_{j=1}^{k} A_n \times A_m}{\left(\sum_{j=1}^{k} A_n^2 + \sum_{j=1}^{k} A_m^2 - \sum_{j=1}^{k} A_n \times A_m\right)} \end{aligned} \tag{10}$$

The $\mathcal{I}_{SS}(A_m)$ function determines the structural similarity among the target A_n and A_m, the higher the numerical value is, a more closer structural description of A_m instance is with A_n description [31,32]. As a result, structural attributes are suggested for a tuple A_m with missing attributes.

3.5.2. Instance-Based Imputation Measure

The ABox $\mathcal{A}$ is comprised of a finite set of membership assertions $\mathcal{A}$ referring to the concepts and membership roles to their respective TBox $\mathcal{T}$. The set of assertions $\mathcal{A}$ for the *UCamI* dataset is represented as:

$$\mathcal{A} \leftarrow (ts, r_s, \mathcal{R}_i, \mathcal{V}_i) \tag{11}$$

Each of the assertion is a combination of sensors r_s with their certain states $\mathcal{V}_i$ at a timestamp ts.

$$(r_s, \mathcal{R}_i, \mathcal{V}_i) \leftarrow \langle binsens_{1\ldots30}, \mathcal{R}_\alpha, \mathcal{V}_\alpha \rangle \bigcup \langle ble_{1\ldots15}, \mathcal{R}_\beta, \mathcal{V}_\beta \rangle \bigcup \langle c_{1\ldots8}, \mathcal{R}_\epsilon, \mathcal{V}_\epsilon \rangle \bigcup \langle acc_{x,y,z}, \mathcal{R}_\varphi, \mathcal{V}_\varphi \rangle \tag{12}$$

where $binsens_{1\ldots30}$ are the object names referring to the concept *BinarySensor* in the *SemImputOnt*, ranging from 1 and 30 with binary states $[0, 1]$ represented as $\mathcal{V}_\alpha$. $ble_{1\ldots15}$ refers object names, which are members for *Proximity* concept having values $\mathcal{V}_\beta$, *Intelligent Floor* concept having assertions $c_{1\ldots8}$ with values $\mathcal{V}_\epsilon$ and accelerometer *SmartWatch* concept having membership for with values as $\mathcal{V}_\varphi$. Instance-based similarity $\mathcal{I}_{SI}(A_m)$ is measured [33] between target activity instance $\mathcal{A}_n$ and instance with missing states $\mathcal{A}_m$ as:

$$\begin{aligned} \mathcal{I}_{SI}(A_m) &= Sim_I(\mathcal{A}_n, \mathcal{A}_m) \\ &= max_m \frac{overlap\,(\mathcal{A}_n, \mathcal{A}_m, m)}{\mathcal{A}_n \uplus \mathcal{A}_m} \end{aligned} \tag{13}$$

where m is the mapping between $\mathcal{A}_n$ and $\mathcal{A}_m$ in conjunction with concept-to-concept and roles-to-roles. In addition, $\mathcal{A}_n \uplus \mathcal{A}_m$ represents the disjoint union of memberships pertaining to concepts and their roles between them. Instance-based similarity exploits neighborhood similarity by measuring similarity through $Sim_I(\mathcal{A}_n, \mathcal{A}_m)$ function. Thus, an instance with high similarity value is chosen for attribute states to be imputed for a tuple $\mathcal{A}_m$ with missing states.

3.5.3. Longitudinal Imputation Measure

The quality of data, resulting from structure and instance-based imputation in a matrix form, is further improved by using classical techniques of Last Observation Carried Forward (LOCF) and Next Observation Carried Backward (NOCB). LOCF and NOCB are applied to the data in an observable manner by analyzing each longitudinal segment, as described in Equation (7), for activity states retrieved through SPARQL queries. While observing the binary sensors and their states in a time series longitudinal segments, it is observed that the sensor states are triggered once either for activation or deactivation. For example, an object *Washing Machine* in *UCamI* dataset has a *contact* type sensor with *Open* state at T_1 = 2017-11-10 13:37:56.0 and *Close* state at T_2 = 2017-11-10 13:38:39.0. In this case, while synchronizing this sensor data with other states per unit time, *Null* values appear after T_1 till T_2 as the states triggered for once. For this LOCF, a *sample-and-hold* method is activated, which carries forward the last state and imputes the *Null* values with this last available sensor state. Similarly, NOCB imputes the missing values from next available state, which is carried backwards. The missing states for Proximity sensors in the case of the *UCamI* dataset are imputed in a slightly different way as elaborated in Algorithm 1. It identifies the proximity sensors and their respective RSSI values within the sliding window. The proximity sensor utilizes maximum value imputation in which the LOCF method is applied until some other proximity sensor with a value greater than the already known value is identified. For continuous data such as *Floor* and *Acceleration*, a statistical approach is adopted to replace the missing states with the mean of corresponding observed attributes. Mean imputation method tends to be robust and easy to substitute the missing values.

3.6. Classification

To cross examine the effectiveness for imputed datasets using proposed *SemImput* framework, we used a Deep Learning-based Artificial Neural Network (*ANN*) classifier [34]. The experimental results proved to be suitable for multimodal, multi-sensory, and multi-feature datasets for HAR. For this, an *ANN* model is trained with the labeled 2D training matrix instances for the *UCamI*, *Opportunity* and *UCI-ADL* datasets. The computational complexity and recognition accuracies are then assessed.

3.6.1. One-Hot Code Vectorization

It has been observed as advantageous to transform categorical variables using suitable feature engineering before applying neural network [35]. For this, we used *one-hot* encoding, a robust feature engineering scheme, for generating the suitable feature vector indices [16]. These categorical features are mapped into sensor state vector indices representing the concurrent sensor activation patterns for a particular activity. This scheme expands the dimension of the feature matrix for 2^n possible combinations based on the binary states for the "n" sensors involved in the feature vector. As described in Algorithm 2, n-dimensional sparse vector per unit time is obtained for populating feature matrix required for classification. The value 1 is encoded where the sensor has an active state and the value 0 is assigned for *missing* state in a row vector [35]. The missing value indicator r in the matrix is represented as $r_{n,p}$ with n_{th} row and p_{th} column:

$$r_{n,p} = \begin{cases} 1, & \text{value is observed} \\ 0, & \text{if value is missing} \end{cases} \tag{14}$$

3.6.2. Artificial Neural Networks for HAR

We introduced a Semantic Deep Learning-based Artificial Neural Network (*SemDeep-ANN*) having the ability to extract hierarchy of abstract features [36,37] using a stack of convolutional operators, which are supported by Convolutional Neural Networks (*CNN*). *SemDeep-ANN* consists of three layers namely *input layer*, *hidden layers*, and *output layer*, which use vectorized data to train model for probability estimation over the test data. The estimated probabilities are obtained from the output

layer through the *soft_max* activation function in addition to gradient descent algorithm. Further details of the *SemDeep-ANN* are given in Algorithm 3.

Algorithm 2 Semantic Vectorization Using One-Hot Coding Technique

Input: A_m^{Imp} ▷ Extract scalar sequence (BinSens, Proximity)

Output: M ▷ Vectorized feature Matrix.

procedure SEMANTICVECTORIZATION
  for all timestamp t = 1 to T **do**
    function $BinSensVectorization(CQ, A_m^{Imp})$
      for $(cq_i \epsilon \mathcal{CQ})$ **do**
        $BinSens_{Attrib} \leftarrow execute(cq_i).filter(BinSens, A_m^{Imp}))$ ▷ using SPARQL Queries
        $BinSens_{states} \leftarrow execute(cq_i).filter(BinSens_{Attrib}))$
        while $BinSens_{states} \neq \phi$ **do**
          $BinSens_{Vec} \leftarrow Map(BinSens, BinSens_{Attrib})$
          $BinSens_{fCol} \leftarrow Transform(n \times p, BinSens_{Vec})$ ▷ transform rows into columns
          $BinSens_{stride} \leftarrow StateReplace(BinSens_{Vec})$ ▷ 1 for Active *BinSens* or 0, otherwise
      Return $BinSens_{stride}$
    function $ProxVectorization(CQ, A_m^{Imp})$
      for $(cq_i \epsilon \mathcal{CQ})$ **do**
        $Prox_{Attrib} \leftarrow execute(cq_i).filter(Prox, A))$ ▷ using SPARQL Queries
        $Prox_{states} \leftarrow execute(cq_i).filter(Prox_{Attrib}))$
        while $Prox_{states}(state) \neq \phi$ **do**
          $Prox_{Vec} \leftarrow Map(Prox, Prox_{Attrib})$
          $Prox_{fCol} \leftarrow Transform(n \times p, Prox_{Vec})$ ▷ transform rows into columns
          $Prox_{stride} \leftarrow StateReplace(Prox_{Vec})$ ▷ Set 1 for highest RSSI and 0 for rest
      Return $Prox_{stride}$
  $M \leftarrow BinSens_{stride} \parallel Prox_{stride} \parallel A_{floor} \parallel A_{Acc}$
  increment t by 3 sec

Algorithm 3 Semantic Deep Learning-based Artificial Neural Network (SemDeep-ANN)

Input: Labeled Dataset $\mathcal{M}_{lab}$,Unlabeled Dataset $\mathcal{M}_{unlab}$, and labels ▷ Scalar sequence Equation (8)

Output: Activity Labels A_n for the $\mathcal{M}_{unlab}$ ▷ HAR.

procedure DEEP LEARNING HAR
  Forward Propagation
  for all timestamp t = 1 to T **do** ▷ Sliding Widow Process
    $D_F \leftarrow \mathcal{M}_{lab}$ ▷ Retrieve Data (Feature Vectors Matrix)
    $x \leftarrow normalize(D_F)$ ▷ Preprocessing, reordering, filtering examples with no missing labels
    Sample, Split, FE, TV
    Initialize random weights $\{w_1, w_1, \ldots w_n\}^T$ and biasness b
    $y = \sigma\left(\sum_{k=1}^{n} w_k x_k + b\right)$ ▷ applying nonlinear transformation σ using $y = \sigma\left(w^T x + b\right)$
    $fc_y \leftarrow fully\_connected\_NN(y)$
    $A_n \leftarrow soft\_max(fc_y)$ ▷ Update weights in the network
    Backward Propagation
    Compute Cross entropy gradient ▷ Use trained network to predict Activity labels
    Apply gradient descent ▷ Update network parameters
  Activity Labels ← Use trained network model ▷ Predict labels

4. Results and Discussion

The performance evaluation for *SemImput* framework is measured using non-imputed and semantically imputed HAR datasets. The results are compared with other popular methods, which were investigated using the same datasets.

4.1. Data Description

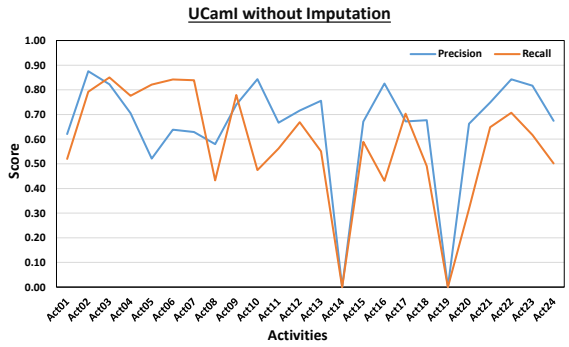
To compare the HAR performance of the proposed *SemImput* framework, firstly, the experiments were performed on the *UCamI* dataset. It offers recognition of 24 set of activities for non-imputed and imputed datasets. Secondly, the *Opportunity* dataset contains manipulative gestures of short duration such as *opening* and *closing*, of *Doors*, *Dishwasher*, and *Drawers*. These were collected for four subjects who were equipped with five different body attached sensors for the tracking of static and dynamic activities [38]. Due to the involvement of several sensors, data transmission problems among wireless sensors lead to segments of data being missed represented by *Null*. For this reason, we analyzed the data and performed the required imputation in order to complement the missing segments of data [37,39]. Lastly, we tested *SemImput* framework on the UCI-ADL dataset, which was collected while monitoring 10 different ADLs [40] using *passive infrared*, *reed switches*, and *float sensors*. These sensors were used to detect *motion*, *opening* and *closing* binary states of the objects and activities such as *toileting*, *sleeping*, *Showering*.

4.2. Performance Metrics

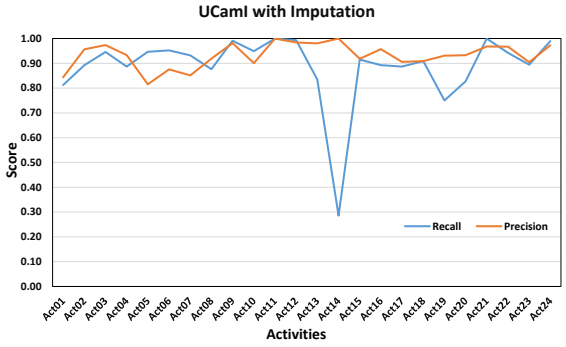
We measured the impact of imputation against the non-imputed datasets using commonly used metrics, such as accuracy, precision, and f-measure. The *SemDeep-ANN* models were validated by splitting the datasets independently into train and test sets using a *leave one day out* approach. During the evaluation process, we retained one full day from each of the dataset for testing, whereas the remaining samples are used as a training set. This process is repeated for each day, with the overall average accuracy obtained as a performance measure.

4.3. Discussion

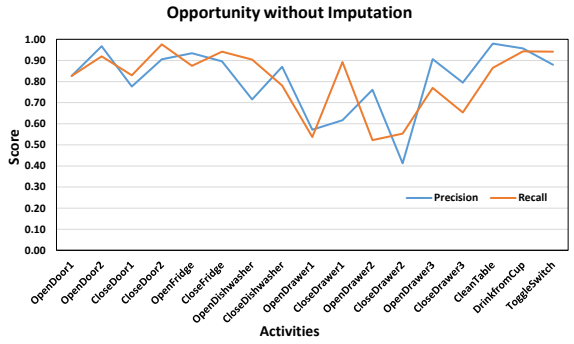
This study examines and evaluates the *SemImput* framework for HAR classification results for which the precision and recall curves are shown in Figure 4a–h. The framework achieved an overall accuracy of 71.03% for set of activities recognized from non-imputed *UCamI* dataset as mentioned in Table 2. The activity *Prepare breakfast* (Act02) yielded the highest precision of 87.55%, but it was also misclassified with the activities *Breakfast* (Act05) and *Dressing* (Act22) respectively. Similarly, the activity *Enter the Smartlab* (Act10) was also classified with the highest precision, it was, however, misclassified as the activity *Put waste in the bin* (Act15). The activity *Breakfast* (Act05) with the lowest precision 52.14% was mostly misclassified as activities *Prepare breakfast* (Act02) and *Wake up* (Act24). Furthermore, the activity *Eat a Snack* (Act08) with lower precision of 57.95% was misclassified as the activity *Prepare Lunch* (Act03) due to the involvement of similar sensors and floor area. The activity *Visit in the SmartLab* (Act14) and *Wash dishes* (Act19) was hard to detect as they have lessor number of annotated examples. The experimental results indicate an increased recognition accuracy to 92.62% after modeling the *UCamI* dataset into ontology-based complex activity structures and by performing the semantic imputation as shown in Figure 4b. The plot for these illustrates that the activity *Breakfast* (Act05) having the lowest recognition precision of 81.54% was most often classified as the activity *Prepare breakfast* (Act02). The activities *Play a videogame* (Act11) and *Visit in the SmartLab* (Act14) were recognized with 100% accuracy, which were having lower accuracies with the non-imputed data. Similarly, the activity *Relax on the sofa* (Act12) was also recognized with the highest precision rate of 98.44% as shown in Table 2. This suggests that semantic data imputation provided positive data values, which resulted in the increase of classification accuracies for individual activities.



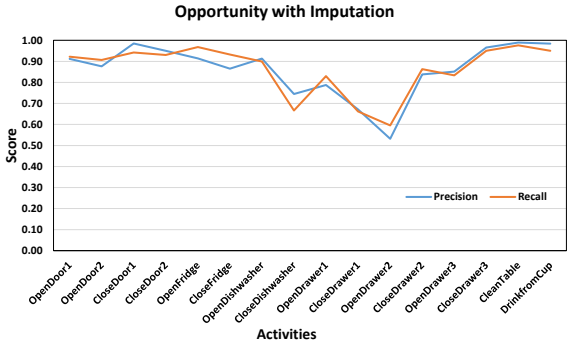
(a) Non-imputed *UCamI* dataset



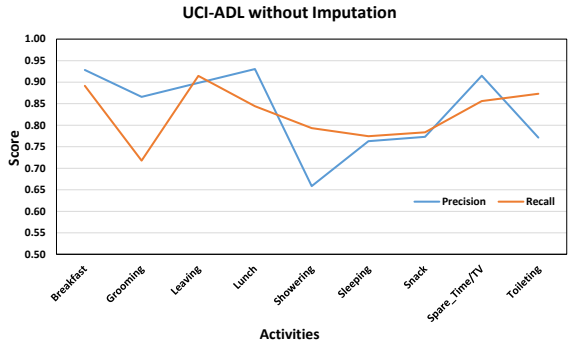
(b) Imputed *UCamI* dataset



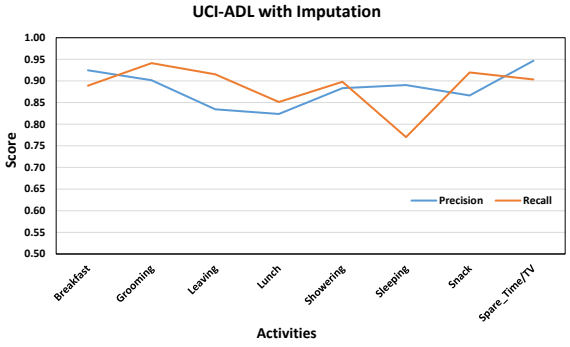
(c) Non-imputed *Opportunity* dataset



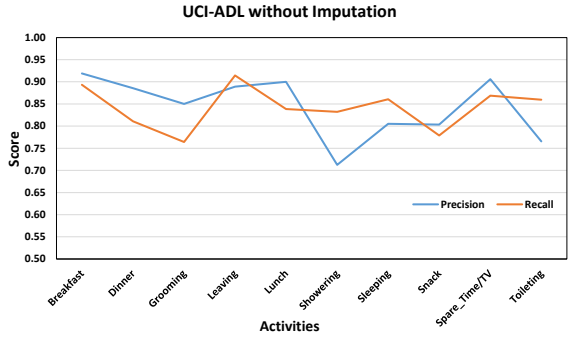
(d) Imputed *Opportunity* dataset



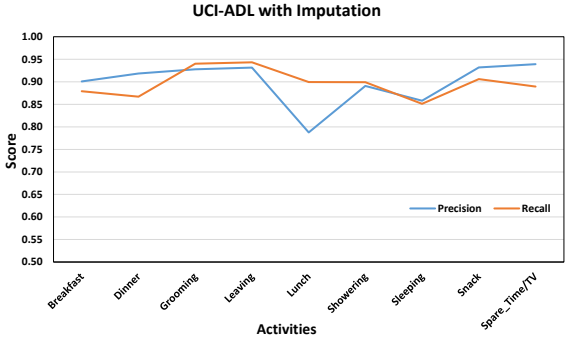
(e) Non-imputed *UCI-ADL* (OrdóñezA) dataset



(f) Imputed *UCI-ADL* (OrdóñezA) dataset



(g) Non-imputed *UCI-ADL* (OrdóñezB) dataset



(h) Imputed *UCI-ADL* (OrdóñezB) dataset

Figure 4. Classification performance of SemImput framework: Precision & Recall.

Table 2. Confusion matrix for per-class HAR using non-imputed & imputed UCamI dataset.

(**Non-imputed**)

Predicted Activities \ Ground Truth Activities	Act_1	Act_2	Act_3	Act_4	Act_5	Act_6	Act_7	Act_8	Act_9	Act_{10}	Act_{11}	Act_{12}	Act_{13}	Act_{14}	Act_{15}	Act_{16}	Act_{17}	Act_{18}	Act_{19}	Act_{20}	Act_{21}	Act_{22}	Act_{23}	Act_{24}
Act_1	62.07	0	0.69	4.48	0	0	5.52	0	0	0.69	0	0	0	0	13.79	0	2.07	0	1.38	2.41	0	6.90	0	0
Act_2	0	87.55	0	0	4.06	0	0	0	0	0	0	0	0.14	0	0	0.14	0.70	0.14	0	2.10	0.42	4.48	0	0.28
Act_3	0.67	0	82.23	0	0	3.04	0	3.88	1.07	1.69	0	0.67	0	0	2.81	0	0.73	0.90	0.39	1.80	0	0.11	0	0
Act_4	5.70	0	0	70.50	0	0	7.43	0	0	3.76	0.61	0	0	0	6.71	0	2.75	0.51	0.61	0	0	1.42	0	0
Act_5	0	10.25	0	0	52.14	0	0	0	0	0	0	0	0	0	0	2.29	8.66	3.08	0	1.00	9.05	2.29	0	11.24
Act_6	0	0	9.43	0	0	63.85	0	1.38	9.92	1.28	0	5.89	0	0	2.95	0	2.46	0.49	0.59	1.38	0	0.39	0	0
Act_7	5.89	0	0	13.72	0	0	62.89	0	0	2.12	4.24	0	0	0	4.51	0	2.12	0	0.18	0	0	2.95	1.38	0
Act_8	1.54	0	11.28	0	0	4.62	0	57.95	4.62	0	0	0	0	0	3.59	0	1.03	3.08	0	7.69	0	4.62	0	0
Act_9	0.31	0	7.28	0	0	2.29	0	1.87	74.01	1.77	0	2.81	0.94	0.31	2.91	0	3.01	1.35	0.10	0.42	0	0.62	0	0
Act_{10}	0	0	2.23	0	0	0	0	0	0	84.36	0	0	0	2.23	11.17	0	0	0	0	0	0	0	0	0
Act_{11}	0	0	0	2.33	0	0	10.08	0	0	0.78	66.67	0	0	0	6.20	0	6.20	0.78	1.55	0	0	4.65	0.78	0
Act_{12}	0.34	0	10.14	0	0	0.34	0	0	10.14	1.69	0	71.62	0	0	3.72	0	1.01	1.01	0	0	0	0	0	0
Act_{13}	0	9.30	0	0	0	1.16	0	0	1.16	0	0	0	75.58	1.16	8.14	0	1.16	0	0	0	1.16	1.16	0	0
Act_{14}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Act_{15}	2.33	0	2.20	3.76	0	2.33	2.20	2.20	1.55	3.63	0.26	0.52	5.18	0.78	67.10	0	1.94	0.39	0.39	0.39	0	2.72	0.13	0
Act_{16}	0	0.76	0	0	0	0	0	0	0	0	0	0	0	0	0	82.58	9.85	1.52	0	0	0	2.27	0	3.03
Act_{17}	0.51	0.68	0.51	0.51	1.70	1.10	0.59	1.44	2.55	0.34	1.02	0.17	0	0	2.97	8.50	67.20	3.74	0.08	0.17	0.34	4.93	0.34	0.59
Act_{18}	0	0	2.49	0	0.50	1.49	0	1.99	0	2.99	0	0	0	0	0.50	1.49	11.94	67.66	1.00	5.47	0	1.99	0	0.50
Act_{19}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Act_{20}	1.25	1.25	6.25	0	2.50	1.25	0	11.25	0	0	0	0	0	0	2.50	0	3.75	0	2.50	66.25	0	1.25	0	0
Act_{21}	0	8.49	0	0	9.27	0	0	0	0	0	0	0	0	0	0	1.93	1.54	0.39	0	0	74.90	1.16	0	2.32
Act_{22}	0.27	2.01	0	0	1.34	0	0.40	0	0	0.13	0.13	0	0.40	0	0.81	0.40	4.83	0.81	0	0.13	0.27	84.30	2.01	1.74
Act_{23}	0	0	0	0	0	0	2.82	0	0	0	0	0	0	0	1.41	0	2.82	1.41	0	0	0	9.86	81.69	0
Act_{24}	0	2.75	0	0	12.84	0	0	0	0	0	0	0	0	0	0	4.13	3.21	1.38	0	0	1.83	6.42	0	67.43

(**Imputed**)

Predicted Activities \ Ground Truth Activities	Act_1	Act_2	Act_3	Act_4	Act_5	Act_6	Act_7	Act_8	Act_9	Act_{10}	Act_{11}	Act_{12}	Act_{13}	Act_{14}	Act_{15}	Act_{16}	Act_{17}	Act_{18}	Act_{19}	Act_{20}	Act_{21}	Act_{22}	Act_{23}	Act_{24}
Act_1	84.38	0	1.20	0.90	0	0.30	6.01	0	0	0	0	0	0	0	3.60	0	1.50	0	1.20	0	0	0.90	0	0
Act_2	0	95.66	0	0	3.39	0	0	0	0	0	0	0	0	0	0	0.14	0.54	0.14	0	0	0	0.14	0	0
Act_3	0.12	0	97.36	0	0	1.50	0	0	0	0	0	0	0	0	0.18	0	0.78	0	0.06	0	0	0	0	0
Act_4	1.76	0	0	93.31	0	0	3.99	0	0	0.12	0	0	0	0	0.59	0	0.23	0	0	0	0	0	0	0
Act_5	0	10.65	0	0	81.54	0	0	0	0	0	0	0	0.13	0	0	0	6.87	0	0	0	0	0.81	0	0
Act_6	0.24	0	9.49	0	0	87.54	0	0	0	0	0	0	0.12	0	0.24	0	2.25	0	0.12	0	0	0	0	0
Act_7	4.27	0	0	9.45	0	0	85.15	0	0	0	0	0	0	0	1.01	0	0.11	0	0	0	0	0	0	0
Act_8	0	0	0	0	0	0	0	91.90	0	0	0	0	0	0	3.24	0	0	0	0	4.45	0	0.40	0	0
Act_9	0	0	0	0	0	0	0	0	98.16	0	0	0	0	1.08	0	0	0	0.76	0	0	0	0	0	0
Act_{10}	0	0	0.90	2.10	0	0	0	0	0	90.09	0	0	0	0	5.41	0	1.50	0	0	0	0	0	0	0
Act_{11}	0	0	0	0	0	0	0	0	0	0	100.0	0	0	0	0	0	0	0	0	0	0	0	0	0
Act_{12}	0	0	0	0	0	0	0	0	0	0	0	98.44	0.94	0	0	0	0.31	0	0	0	0	0.31	0	0
Act_{13}	0	0	0	0	0	0	0	0	0	0	0	0	98.06	0	0	0	0	0.97	0	0	0	0.97	0	0
Act_{14}	0	0	0	0	0	0	0	0	0	0	0	0	0	100.0	0	0	0	0	0	0	0	0	0	0
Act_{15}	0.34	0	0.23	0.45	0	0.11	0	2.16	0.23	1.48	0	0.23	0.11	0	91.82	0	0.91	0.45	0.34	0.91	0	0.23	0	0
Act_{16}	0	0.43	0	0	0	0	0	0	0	0	0	0	0	0	0	95.73	2.14	1.28	0	0	0	0	0	0.43
Act_{17}	0.36	0.18	0.36	0.27	0.54	0.91	0.09	0.18	0.36	0.18	0	0	0.91	0	1.27	1.00	90.55	0.73	0	0.18	0	1.91	0	0
Act_{18}	0	0.36	0	0	0	0	0	0.36	0.73	0	0	0	0	0	0	4.74	0.73	90.88	0	1.09	0	0.36	0	0.73
Act_{19}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6.90	0	0	0	93.10	0	0	0	0	0
Act_{20}	0	0	0	0	0	0	0	6.71	0	0	0	0	0	0	0	0	0	0	0	93.29	0	0	0	0
Act_{21}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.32	1.28	0	0	0	96.79	1.60	0	0
Act_{22}	0.12	0	0	0	0.35	0	0	0	0	0	0	0	0.12	0	0.12	0	0.81	0	0	0.58	0	96.74	1.16	0
Act_{23}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1.06	0	0	0	0	0	0	8.51	90.43	0
Act_{24}	0	0.67	0	0	0	0	0	0	0	0	0	0	1.00	0	0	0.33	0	0.33	0	0	0	0.33	0	97.32

The *Opportunity* dataset represents 17 ADLs and is of complex nature by having missing samples labeled as *Null* due to sensor disconnections. Figure 4c,d shows the per class precision and recall for recognized ADLs with the *Opportunity* dataset. The presented framework evaluates the *Opportunity* dataset without the 'Null' class by obtaining an overall accuracy of 86.57%, and an increased accuracy with the imputed dataset by 91.71%. The comparisons for both confusion matrices are shown in Table 3.

As shown in Figure 4e,f for the *UCI-ADL* Ordóñez-A raw dataset, an overall classification result with 82.27% accuracy was obtained. It included activities like *Grooming*, *Spare_Time/TV*, and *Toileting* having the most number of instances and the activity *Lunch* with minimum number of instances. However, the classification results as mentioned in Table 4 show that the activities *Leaving* and *Breakfast* have the highest recognition accuracy as compared to the activity *Grooming* with the lower classification accuracy. In order to verify the proposed *SemImput* framework, it was also tested on the semantically imputed *UCI-ADL* Ordóñez-A dataset. This resulted in an increased recognition accuracy for activities such as *Breakfast*, *Lunch*, and *Leaving* significantly as shown in Figure 4f. It was due to the introduction of the semantic structure understanding of events with respect to morning, afternoon, and generalization of semantic rules for such activities for imputing missing values. The improvement in statistical quality through imputation raised the recognition accuracy significantly up

to 89.20%. Similarly, an increased performance is also observed for the *UCI-ADL* Ordóñez-B dataset for the overall activities with imputed data, especially for the *Dinner* and *Showering* as shown in Table 5. The global accuracy for *UCI-ADL* Ordóñez-B dataset was improved from 84.0% to 90.34%, which also proves the significance of proposed framework as shown in Table 6.

Table 3. Confusion matrix for per-class HAR using non-imputed & imputed Opportunity dataset.

(**Non-imputed**)

Predicted Activities \ Ground Truth Activities	*OpenDoor1*	*OpenDoor2*	*CloseDoor1*	*CloseDoor2*	*OpenFridge*	*CloseFridge*	*OpenDishwasher*	*CloseDishwasher*	*OpenDrawer1*	*CloseDrawer1*	*OpenDrawer2*	*CloseDrawer2*	*OpenDrawer3*	*CloseDrawer3*	*CleanTable*	*DrinkfromCup*	*ToggleSwitch*
OpenDoor1	82.65	0	16.33	0	0	0	0	0	0	0	0	0	0	0	0	0	1.02
OpenDoor2	0	91.98	0.62	7.41	0	0	0	0	0	0	0	0	0	0	0	0	0
CloseDoor1	17.05	0	82.95	0	0	0	0	0	0	0	0	0	0	0	0	0	0
CloseDoor2	0	1.57	0.79	97.64	0	0	0	0	0	0	0	0	0	0	0	0	0
OpenFridge	0.26	0.26	0	0	87.47	7.42	2.81	0.51	0	0.26	0	0	0	0	0.26	0.26	0.51
CloseFridge	0	0	0	0	3.65	94.16	0	0.36	0.36	0.73	0	0	0	0	0	0.36	0.36
OpenDishwasher	0	0	0	0	2.40	0	90.42	2.99	0.60	0	0	0	0	1.20	0	1.20	1.20
CloseDishwasher	0	0	0	0	2.92	0	10.95	78.10	0	0	0	1.46	0	4.38	0	0.73	1.46
OpenDrawer1	0	0	0	0	0	0	1.49	2.99	53.73	25.37	0	1.49	0	2.99	0	0	11.94
CloseDrawer1	0	1.35	0	0	1.35	0	0	0	6.76	89.19	1.35	0	0	0	0	0	0
OpenDrawer2	0	0	0	0	1.49	0	1.49	0	19.40	11.94	52.24	4.48	1.49	0	0	0	7.46
CloseDrawer2	0	0	0	0	2.13	0	0	8.51	4.26	21.28	6.38	55.32	2.13	0	0	0	0
OpenDrawer3	0	0	0	0	0	0	4.42	0	1.77	0	6.19	3.54	76.99	6.19	0	0	0.88
CloseDrawer3	0	0	0	0	0	0	0	0.99	0	0	0	26.73	6.93	65.35	0	0	0
CleanTable	0	0	0	0	1.18	0.59	1.76	0	0	0	0	0	0	0	86.47	10.00	0
DrinkfromCup	0.18	0.18	0.37	0.18	0	0	4.40	0.18	0	0	0	0	0	0	0.18	94.32	0
ToggleSwitch	0	0	0.58	0	0.58	0	0	0	1.75	1.75	0	0	0	0	0.58	0.58	94.15

(**Imputed**)

Predicted Activities \ Ground Truth Activities	*OpenDoor1*	*OpenDoor2*	*CloseDoor1*	*CloseDoor2*	*OpenFridge*	*CloseFridge*	*OpenDishwasher*	*CloseDishwasher*	*OpenDrawer1*	*CloseDrawer1*	*OpenDrawer2*	*CloseDrawer2*	*OpenDrawer3*	*CloseDrawer3*	*CleanTable*	*DrinkfromCup*	*ToggleSwitch*
OpenDoor1	90.00	0	10.00	0	0	0	0	0	0	0	0	0	0	0	0	0	0
OpenDoor2	0	92.26	0	1.19	0	0	0	0	0.60	1.79	0	1.19	2.98	0	0	0	0
CloseDoor1	1.16	0	90.70	0	1.16	1.16	5.81	0	0	0	0	0	0	0	0	0	0
CloseDoor2	0	3.62	0	94.20	0	0	0	0.72	0.72	0.72	0	0	0	0	0	0	0
OpenFridge	0.52	0	0	0	93.04	5.67	0.52	0	0	0	0	0	0	0	0	0	0.26
CloseFridge	0	0.79	0	0	1.98	96.84	0	0.40	0	0	0	0	0	0	0	0	0
OpenDishwasher	0.56	0	0	0	3.35	0	93.30	2.23	0.56	0	0	0	0	0	0	0	0
CloseDishwasher	0	0	0	0	1.55	0	7.75	89.92	0	0	0	0.78	0	0	0	0	0
OpenDrawer1	0	1.75	0	0	0	0	5.26	1.75	66.67	10.53	3.51	1.75	5.26	1.75	0	0	1.75
CloseDrawer1	1.06	1.06	1.06	0	1.06	0	5.32	0	1.06	82.98	0	0	0	1.06	3.19	2.13	0
OpenDrawer2	0	1.41	0	0	5.63	0	0	1.41	8.45	2.82	66.20	8.45	1.41	0	0	2.82	1.41
CloseDrawer2	0	4.76	0	0	0	0	0	4.76	2.38	11.90	9.52	59.52	0	7.14	0	0	0
OpenDrawer3	0	2.94	0	0	0	0	0	0	0	0	7.84	0	86.27	2.94	0	0	0
CloseDrawer3	0	0	0	0	0	0	0	0	0	0	2.08	9.38	5.21	83.33	0	0	0
CleanTable	0	0	0	0	0	0	0	0	0	0	2.22	1.67	0	0	95.00	1.11	0
DrinkfromCup	0.17	0	0	0	0	0	0.17	0.17	0	0.34	0.51	0	0	0.51	0.51	97.62	0
ToggleSwitch	0	0	0	0	0	0	0	0	1.00	1.00	0	0	1.49	1.49	0	0	95.02

Table 4. Confusion matrix for per-class HAR using non-imputed & imputed *UCI-ADL* (OrdóñezA) dataset.

(**Non-imputed**)

Predicted Activities \ Ground Truth Activities	*Breakfast*	*Grooming*	*Leaving*	*Lunch*	*Showering*	*Sleeping*	*Snack*	*Spare_Time/TV*	*Toileting*
Breakfast	89.12	0.20	2.20	0.74	3.21	3.92	0	0.51	0.10
Grooming	2.46	71.00	9.21	2.30	5.26	6.67	2.01	0.62	0.47
Leaving	0.23	0.02	91.12	0.23	0.32	2.23	2.73	3.02	0.10
Lunch	3.20	0.09	0.10	84.41	3.62	4.61	3.26	0.12	0.59
Showering	0.01	5.30	0	0	79.12	4.33	8.64	0.32	2.28
Sleeping	0	4.49	0.01	0	5.21	77.15	6.23	0.20	6.71
Snack	0.12	0.02	0.04	3.32	12.54	0.02	77.50	3.20	3.24
Spare_Time/TV	0.05	0.10	0	0.02	4.32	0.88	0	85.10	9.54
Toileting	1.20	1.05	0.18	0.12	8.12	1.20	0	0	88.13

(**Imputed**)

Predicted Activities \ Ground Truth Activities	*Breakfast*	*Grooming*	*Leaving*	*Lunch*	*Showering*	*Sleeping*	*Snack*	*Spare_Time/TV*	*Toileting*
Breakfast	96.51	0.21	0.35	1.21	0.32	0.01	0.04	0.02	1.33
Grooming	0.12	88.01	7.20	3.39	0.83	0.04	0.18	0	0.23
Leaving	0.14	0.21	94.02	0	0.50	1.79	3.10	0.13	0.11
Lunch	1.21	4.45	1.65	91.12	0.75	0.41	0.03	0.38	0
Showering	0	0.62	0.75	2.10	85.23	2.20	6.11	2.98	0.01
Sleeping	0.87	0.55	0.02	5.45	4.39	88.69	0.01	0.02	0
Snack	0.01	0.74	0	5.64	2.21	6.88	77.02	7.01	0.49
Spare_Time/TV	0	0.43	0	1.20	1.35	0.35	0.40	92.45	3.82
Toileting	1.01	0	0	0	8.60	0	0	1.32	89.07

Table 5. Confusion matrix for per-class HAR using non-imputed & imputed *UCI-ADL* (OrdóñezB) dataset.

(**Non-imputed**)

Predicted Activities \ Ground Truth Activities	*Breakfast*	*Dinner*	*Grooming*	*Leaving*	*Lunch*	*Showering*	*Sleeping*	*Snack*	*Spare Time/TV*	*Toileting*
Breakfast	88.95	1.65	0	0.01	3.45	0.06	0.08	4.21	1.36	0.23
Dinner	0.64	81.06	1.23	0.98	4.55	0.16	1.88	7.26	2.24	0
Grooming	0.35	0.12	76.43	0.21	0	1.18	16.23	0.45	4.61	0.42
Leaving	0	0.02	0.29	91.49	0.36	0.12	0	0.30	3.10	4.32
Lunch	2.01	1.71	2.36	0.92	83.90	0.26	0.08	4.68	3.45	0.63
Showering	0.83	1.65	1.65	3.70	0	83.17	0.61	0.84	0.12	7.43
Sleeping	2.34	4.65	3.87	4.62	0.15	2.64	81.73	0	0	0
Snack	1.60	0.65	0.23	0	0.45	0	0	77.92	5.75	13.40
Spare Time/TV	0	0	2.89	0.54	0.03	6.25	0	0	90.29	0
Toileting	0.03	0	1.01	0.50	0.32	8.16	0.90	0	3.08	86.00

(**Imputed**)

Predicted Activities \ Ground Truth Activities	*Breakfast*	*Dinner*	*Grooming*	*Leaving*	*Lunch*	*Showering*	*Sleeping*	*Snack*	*Spare Time/TV*	*Toileting*
Breakfast	97.10	0.01	0.20	0.12	0	0.65	0.32	1.41	0	0.19
Dinner	0.80	87.90	1.65	0.65	1.32	0.53	2.64	3.21	1.21	0.09
Grooming	0.32	2.40	86.69	0.07	2.45	2.89	0.05	4.08	0.10	0.95
Leaving	0.01	1.12	0.98	94.01	0.51	0.03	1.32	0.85	0.05	1.12
Lunch	0.45	0.06	0	1.32	94.30	0.12	1.24	1.19	1.32	0
Showering	0.78	1.36	1.40	0.45	0.09	94.47	0.08	0.01	0.24	1.12
Sleeping	0.63	1.89	0	1.61	0.65	3.99	89.87	0.31	0.08	0.97
Snack	1.10	2.45	1.77	1.74	1.11	0.99	2.10	85.10	2.49	1.15
Spare Time/TV	0	0.32	0.65	0.89	0.47	1.54	3.19	2.15	90.58	0.21
Toileting	0.01	0.09	1.05	0.45	0.25	7.14	0.06	0.89	1.11	88.95

Table 6. Recognition accuracy gain using the proposed *SemImput* framework (Unit: %).

Method	Datasets	Number of Activities	(Mean Recognition Accuracy) Non-Imputed	Imputed	Standard Deviation
Proposed *SemImput*	*Opportunity* [20]	17	86.57	91.71	±2.57
	UCI-ADL OrdóñezA [40]	9	82.27	89.20	±3.47
	UCI-ADL OrdóñezB [40]	10	84.0	**90.34**	±3.17
	UCamI [19]	24	71.03	**92.62**	±10.80

As shown in Table 7, the proposed *SemImput* framework along with *SemDeep-ANN* model not only improved the recognition rate for individual activities within the datasets but also improved the global accuracy over each dataset. We also compared the activity classification performance of our framework with a different state-of-the-art methods. The presented results show the potential of *SemImput* framework with significant accuracy gain. Although for the *UCI-ADL* Ordóñez-A and *Opportunity* datasets, our methodology was worse, it still achieved significant recognition performance score of 89.20% and 91.71%, respectively. These findings show that combining the ADLs classification with semantic imputation can lead to comparatively better HAR performance.

Table 7. Comparison results of the proposed *SemImput* framework with state-of-the-art HAR Methods.

State-of-the-Art Methods	Datasets	Number of Activities	Mean Recognition Accuracy(%)	SemImput Gain
Razzaq et al. [22]	*UCamI* [19]	24	47.01	**+45.61**
Salomón et al. [41]	*UCamI* [19]	24	90.65	**+1.97**
Li et al. [37]	*Opportunity* [20]	17	**92.21**	−0.50
Salguero et al. [12,39]	*UCI-ADL* OrdóñezA [40]	9	**95.78**	−6.58
	UCI-ADL OrdóñezB [40]	10	86.51	**+3.83**

5. Conclusions and Future Work

This paper proposed a novel *SemImput* framework to perform *Semantic Imputation* for missing data using public datasets for offline recognition of ADLs. It leverages the strengths of both structure-based and instance-based similarities while performing semantic data imputation. By using ontological model *SemImputOnt*, it uses SPARQL queries executed over the ABox data for semantic data expansion, conjunction separation, identification of missing attributes, and their instances leading towards semantic imputation. In order to further increase the quality of the data, we also utilized time-series longitudinal imputation. The obtained results and presented analysis suggest that gain in recognition accuracy varies with the nature and quality of dataset through the *SemImput*. We validated it, over *UCamI*, *Opportunity*, and UCI-ADL datasets. It achieves the highest accuracy of 92.62% for *UCamI* dataset using a SemDeep-ANN pre-trained model. A substantial, comprehensive, and comparative analysis with state-of-the-art methodologies for these three datasets were also performed and presented

in this paper. Based on the empirical evaluation, it was shown that *DeepSem-ANN* consistently performed well on semantically imputed data by achieving an improved overall classification accuracy. Such a technique can be applied for HAR based systems, which generate data from obtrusive and unobtrusive sources in a smart environment. In the future, we plan to explore, execute, and enhance the *SemImput* framework for real-time HAR systems. Furthermore, we plan to extend our methodology for improving longitudinal imputation as some accuracy degradation is observed while recognizing HAR. We believe that our approach will help in increasing the quality of smart-home data by performing missing data imputation and will increase the recognition accuracy. On the negative side, the *SemImput* framework requires an ontology modeling effort for any activity inclusion or an introduction of a new dataset. For this, we plan to explore a scheme for unified activity modeling ontology for representing the same activities and investigate it further for HAR performance.

Author Contributions: M.A.R. is the principal researchers and main authors of this work. He has initially proposed the idea, implemented, and evaluated the methodology. I.C. has reviewed the initial manuscript and modified representations. S.L. and C.N. have supervised the whole ideation phase and provided discussions for achieving this scientific content. All authors have read and approved the final manuscript.

Acknowledgments: This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2017-0-01629) supervised by the IITP (Institute for Information and communications Technology Promotion), and this work was supported by the Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No. 2017-0-00655) and NRF-2016K1A3A7A03951968 and NRF-2019R1A2C2090504. This work was supported by the REMIND project, which has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sk lodowska-Curie Grant Agreement No. 734355.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

ADL	Activities of Daily Living
HAR	Human Activity Recognition
OWL	Web Ontology Language
SemImput	Semantic Imputation
SemImputOnt	Semantic Imputation Ontology
LOCF	Last Observation Carried Forward
NOCB	Next Observation Carried Backward
SemDeep ANN	Semantic Deep Artificial Neural Network
BLE	Bluetooth Low Energy

References

1. Safyan, M.; Qayyum, Z.U.; Sarwar, S.; García-Castro, R.; Ahmed, M. Ontology-driven semantic unified modelling for concurrent activity recognition (OSCAR). *Multimed. Tools. Appl.* **2019**, *78*, 2073–2104. [CrossRef]
2. Yang, J.; Nguyen, M.N.; San, P.P.; Li, X.L.; Krishnaswamy, S. Deep convolutional neural networks on multichannel time series for human activity recognition. In Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, Buenos Aires, Argentina, 25–31 July 2015.
3. Kautz, T.; Groh, B.H.; Hannink, J.; Jensen, U.; Strubberg, H.; Eskofier, B.M. Activity recognition in beach volleyball using a Deep Convolutional Neural Network. *Data Min. Knowl. Discov.* **2017**, *31*, 1678–1705. [CrossRef]
4. Krishnan, N.C.; Cook, D.J. Activity recognition on streaming sensor data. *Pervasive Mob. Comput.* **2014**, *10*, 138–154. [CrossRef] [PubMed]
5. Chen, J.; Zhang, Q. Distinct Sampling on Streaming Data with Near-Duplicates. In Proceedings of the 37th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, Houston, TX, USA, 10–15 June 2018; ACM: Houston, TX, USA, 2018; pp. 369–382.
6. Chen, L.; Hoey, J.; Nugent, C.D.; Cook, D.J.; Yu, Z. Sensor-based activity recognition. *IEEE Trans. Syst. Man Cybern. C Appl. Rev.* **2012**, *42*, 790–808. [CrossRef]

7. Farhangfar, A.; Kurgan, L.A.; Pedrycz, W. A novel framework for imputation of missing values in databases. *IEEE Trans. Syst. Man Cybern. Syst. Hum.* **2007**, *37*, 692–709. [CrossRef]
8. Farhangfar, A.; Kurgan, L.; Dy, J. Impact of imputation of missing values on classification error for discrete data. *Pattern Recognit.* **2008**, *41*, 3692–3705. [CrossRef]
9. Ni, Q.; Patterson, T.; Cleland, I.; Nugent, C. Dynamic detection of window starting positions and its implementation within an activity recognition framework. *J. Biomed. Inform.* **2016**, *62*, 171–180. [CrossRef]
10. Chernbumroong, S.; Cang, S.; Yu, H. A practical multi-sensor activity recognition system for home-based care. *Decis. Support Syst.* **2014**, *66*, 61–70. [CrossRef]
11. Bae, I.H. An ontology-based approach to ADL recognition in smart homes. *Future Gener. Comput. Syst.* **2014**, *33*, 32–41. [CrossRef]
12. Salguero, A.; Espinilla, M.; Delatorre, P.; Medina, J. Using ontologies for the online recognition of activities of daily living. *Sensors* **2018**, *18*, 1202. [CrossRef]
13. Sarker, M.K.; Xie, N.; Doran, D.; Raymer, M.; Hitzler, P. Explaining Trained Neural Networks with Semantic Web Technologies: First Steps. *arXiv* **2017**, arXiv:cs.AI/1710.04324.
14. Demri, S.; Fervari, R.; Mansutti, A. Axiomatising Logics with Separating Conjunction and Modalities. In Proceedings of the European Conference on Logics in Artificial Intelligence, Rende, Italy, 7–11 May 2019; pp. 692–708.
15. Meditskos, G.; Dasiopoulou, S.; Kompatsiaris, I. MetaQ: A knowledge-driven framework for context-aware activity recognition combining SPARQL and OWL 2 activity patterns. *Pervasive Mob. Comput.* **2016**, *25*, 104–124. [CrossRef]
16. Amador-Domínguez, E.; Hohenecker, P.; Lukasiewicz, T.; Manrique, D.; Serrano, E. An Ontology-Based Deep Learning Approach for Knowledge Graph Completion with Fresh Entities. In Proceedings of the International Symposium on Distributed Computing and Artificial Intelligence, Avila, Spain, 26–28 June 2019; pp. 125–133.
17. Socher, R.; Chen, D.; Manning, C.D.; Ng, A. Reasoning with neural tensor networks for knowledge base completion. In Proceedings of the Advances in neural information processing systems, Lake Tahoe, NV, USA, 5–10 December 2013; pp. 926–934.
18. Zhu, Y.; Ferreira, J. Data integration to create large-scale spatially detailed synthetic populations. In *Planning Support Systems and Smart Cities*; Springer: Cham, Switzerland, 2015; pp. 121–141.
19. UCAmI Cup 2018. Available online: http://mamilab.esi.uclm.es/ucami2018/UCAmICup.html (accessed on 11 March 2020).
20. Opportunity Dataset. Available online: http://www.opportunity-project.eu/challengeDownload.html (accessed on 11 March 2020).
21. ADLs Recognition Using Binary Sensors Dataset. Available online: https://archive.ics.uci.edu/ml/datasets/Activities+of+Daily+Living+%28ADLs%29+Recognition+Using+Binary+Sensors (accessed on 11 March 2020).
22. Razzaq, M.A.; Cleland, I.; Nugent, C.; Lee, S. Multimodal Sensor Data Fusion for Activity Recognition Using Filtered Classifier. *Proceedings* **2018**, *2*, 1262. [CrossRef]
23. Ning, H.; Shi, F.; Zhu, T.; Li, Q.; Chen, L. A novel ontology consistent with acknowledged standards in smart homes. *Comput. Netw.* **2019**, *148*, 101–107. [CrossRef]
24. Okeyo, G.; Chen, L.; Wang, H.; Sterritt, R. Dynamic sensor data segmentation for real-time knowledge-driven activity recognition. *Pervasive Mob. Comput.* **2014**, *10*, 155–172. [CrossRef]
25. Razzaq, M.; Villalonga, C.; Lee, S.; Akhtar, U.; Ali, M.; Kim, E.S.; Khattak, A.; Seung, H.; Hur, T.; Bang, J.; et al. mlCAF: Multi-level cross-domain semantic context fusioning for behavior identification. *Sensors* **2017**, *17*, 2433. [CrossRef]
26. Wan, J.; O'grady, M.J.; O'hare, G.M. Dynamic sensor event segmentation for real-time activity recognition in a smart home context. *Pers. Ubiquitous Comput.* **2015**, *19*, 287–301. [CrossRef]
27. Triboan, D.; Chen, L.; Chen, F.; Wang, Z. A semantics-based approach to sensor data segmentation in real-time Activity Recognition. *Future Gener. Comput. Syst.* **2019**, *93*, 224–236. [CrossRef]
28. Chen, R.; Tong, Y. A two-stage method for solving multi-resident activity recognition in smart environments. *Entropy* **2014**, *16*, 2184–2203. [CrossRef]

29. Zhou, J.; Huang, Z. Recover missing sensor data with iterative imputing network. In Proceedings of the Workshops at the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018.
30. Liu, J.; Li, Y.; Tian, X.; Sangaiah, A.K.; Wang, J. Towards Semantic Sensor Data: An Ontology Approach. *Sensors* **2019**, *19*, 1193. [CrossRef]
31. Yang, A.C.; Hsu, H.H.; Lu, M.D. Imputing missing values in microarray data with ontology information. In Proceedings of the 2010 IEEE International Conference on Bioinformatics and Biomedicine Workshops (BIBMW), Hong Kong, China, 18 December 2010; pp. 535–540.
32. Song, S.; Zhang, A.; Chen, L.; Wang, J. Enriching data imputation with extensive similarity neighbors. *PVLDB Endow.* **2015**, *8*, 1286–1297. [CrossRef]
33. Stuckenschmidt, H. A semantic similarity measure for ontology-based information. In Proceedings of the International Conference on Flexible Query Answering Systems, Roskilde, Denmark, 26–28 October 2009; pp. 406–417.
34. Nweke, H.F.; Teh, Y.W.; Al-Garadi, M.A.; Alo, U.R. Deep Learning Algorithms for Human Activity Recognition using Mobile and Wearable Sensor Networks: State of the Art and Research Challenges. *Expert Syst. Appl.* **2018**, 105, 233–261. [CrossRef]
35. Patricio, C.; Gael, V.; Balázs, K. Similarity encoding for learning with dirty categorical variables. *Mach. Learn.* **2018**, *107*, 1477–1494.
36. Moya Rueda, F.; Grzeszick, R.; Fink, G.; Feldhorst, S.; ten Hompel, M. Convolutional neural networks for human activity recognition using body-worn sensors. *Informatics* **2018**, *5*, 26. [CrossRef]
37. Li, F.; Shirahama, K.; Nisar, M.; Köping, L.; Grzegorzek, M. Comparison of feature learning methods for human activity recognition using wearable sensors. *Sensors* **2018**, *18*, 679. [CrossRef] [PubMed]
38. Peng, L.; Chen, L.; Ye, Z.; Zhang, Y. AROMA: A Deep Multi-Task Learning Based Simple and Complex Human Activity Recognition Method Using Wearable Sensors. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* **2018**, *2*, 74. [CrossRef]
39. Salguero, A.G.; Delatorre, P.; Medina, J.; Espinilla, M.; Tomeu, A.J. Ontology-Based Framework for the Automatic Recognition of Activities of Daily Living Using Class Expression Learning Techniques. *Sci. Program.* **2019**, *2019*. [CrossRef]
40. Ordóñez, F.; de Toledo, P.; Sanchis, A. Activity recognition using hybrid generative/discriminative models on home environments using binary sensors. *Sensors* **2013**, *13*, 5460–5477. [CrossRef]
41. Salomón, S.; Tîrnăucă, C. Human Activity Recognition through Weighted Finite Automata. *Proceedings* **2018**, *2*, 1263. [CrossRef]